REMARKS/ARGUMENTS

Applicants respectfully request that the Examiner consider the entire relied upon reference in order to avoid selecting passages out of context from the reference teachings. For Examiner's convenience, Applicants herein provide an explanation of the reference. Specifically, the Background of the reference along with other sections are discussed to provide the Examiner with context as to the meaning of the word "instruction type," as disclosed by the cited reference.

Rejections 35 U.S.C. §103

Claims 1-3, 5-7, 9-12 and 14 are rejected, under 35 U.S.C. §103(a), as being allegedly unpatentable by Larsen et al, (US Pat No. 5,115,500) (hereinafter Larsen). Applicants respectfully traverse in view of the following.

Independent Claim 1 recites a limitation whereby a corresponding address from a memory unit is used where a plurality of possible meanings are associated with an instruction by the same processor, as claimed. Accordingly, one instruction stored in one location may have multiple attributed meanings by the same processor. The meaning of an instruction is the selection of a function as a result of the instruction decoding (see Instant Application, page 3, lines 15-16). Independent Claim 1 further recites that the concatenation forms an extended instruction that increases the number of instructions in an instruction set, as claimed. Moreover, independent Claim 1 recites executing the extended

instruction, wherein the portion of the corresponding address determines a meaning for the extended instruction from the possible meanings, as claimed.

Larsen discloses that programs may be written in high level language for compilation specifically for the machine which is to run the program, thereby recompiling the program for each new machine (see Larsen, col. 1, lines 15-19). Larsen discloses that this recompilation does not permit incompatible instruction formats to reside in the same machine (see Larsen, col. 1, lines 24-27). Larsen discloses that the object is then to have a memory that allows different machine language formats for two or more different machined types to mix such that all the codes can execute on a single processor without the need to recompile (see Larsen, col. 2, lines 11-14 and col. 4, lines 60-66). Thus, instructions for different machine types, e.g., different processors with different language formats, are being stored in the same memory component.

Larsen further discloses that different locations of a memory component are pre-assigned to different machine languages (see Larsen, col. 5, lines 4-6 and lines 41-44). The machine is designed to decode and execute machine level instructions for two or more different processors (see Larsen, col. 5, lines 27-29). If a portion of a memory component is set aside for "type 2" programs, the decode memory decodes all instructions from that portion according to type 2

decoding rule and instructions fetched from type 1 region are decoded based on type 1 decoding rules (see Larsen, col. 6, lines 30-40).

Thus, instructions are decoded based on the decoding rule type as dictated by the location where the instruction is stored. In other words, each instruction stored has one and only one instruction type, as dictated by the location where it is stored. For example, an instruction stored in location '011' is a type 1 instruction and has only one type (see Larsen, Figure 2). As such, Larsen fails to teach or suggest a corresponding address from a memory unit is used where a plurality of possible meanings are associated with an instruction by the same processor, as claimed.

The rejection asserts that "there are a possibility of a plurality of meanings for each instruction depending on the concatenated address bit. When an instruction is fetched, it has one of two meanings, which are an instruction of type 1 if the lower three order bits aren't '111' and an instruction of type 2 if the lower three order bits are '111.'" It appears that the rejection is equating an instruction type, as disclosed by Larsen, to an instruction meaning, as claimed. Applicants disagree with this interpretation because instruction type 1, as disclosed by Larsen, has one and only one corresponding type which is based on its location where it is stored, e.g., '011' (see Larsen, Figure 2). Similarly, instruction type 2, as disclosed by Larsen, has one and only one corresponding type which is based

on its location where it is stored, e.g., '111' location (see Larsen, Figure 2). Therefore, each stored instruction has one and only one type depending on where it is stored whereas independent Claim 1 recites that an instruction stored in one location has a plurality of meanings, as claimed that correspond to a plurality of functions.

Moreover, Larsen discloses different types of instructions that are decoded based on different decoding rule types as dictated by the location where the instruction is stored. Different types of instructions correspond to instructions executable on different processors, as disclosed by Larsen. Thus, the description of different instruction types, as disclosed by Larsen, fails to teach or suggest a plurality of meanings, as claimed that corresponds to a plurality of functions.

Furthermore, the rejection asserts "that the type 1 and type 2 instructions are executed by the same processor." Applicants disagree. Larsen explicitly discloses that the machine is designed to decode and execute machine level instructions for two or more different processors (see Larsen, col. 5, lines 27-29). Thus, Larsen discloses decoding an instruction of type 1 for execution on processor 1 and decoding an instruction of type 2 for execution on processor 2, hence two or more different processors. Thus, not only does Larsen fail to teach or suggest that a plurality of possible meanings are associated with the instruction by a same processor, as claimed, but Larsen teaches away from the

claimed language by disclosing that the machine is designed to decode and execute machine level instructions for two or more different processors.

Furthermore, the rejection asserts that the instruction set architecture is comprised of both the type 1 and type 2 instruction sets and that with concatenation, the number of instructions increases. The rejection further asserts that type 1 and type 2 instructions are different and that this feature results in an increase in the number of instructions available to the overall instruction set architecture. Applicants disagree with this assertion in view of the following.

As discussed and presented above, a type 1 instruction refers to instructions executable on a type 1 processor and a type 2 instruction refers to instructions executable on a type 2 processor. For example, a first instruction may be stored at '000' address which corresponds to instruction type 1 executable on processor 1. In contrast, a second instruction may be stored at '111' address bit and corresponds to instruction type 2 executable on processor 2 (see Larsen, Figure 2). Therefore, there is a one to one correspondence between each processor and each instruction. In other words the number of instructions is increased by storing additional instructions at different locations of the memory component.

In contrast, independent Claim 1 recites that concatenating the instruction, hence the same instruction, forms an extended instruction, thereby increasing the number of instructions, as claimed. The number of instructions is increased because concatenating a different portion of an address where the instruction is stored to the same instruction provides a different meaning, hence different functionality. Unlike Larsen, there is no need to store an additional instruction in a different location of memory component to increase the number of instructions. Thus, Larsen fails to teach or suggest concatenating the instruction to form an extended instruction that increases a number of instructions in an instruction set, as claimed.

Moreover, Larsen discloses using high order address bits to determine the type of the instruction, hence belonging to type 1 processor versus type 2 processor (see Larsen, Figure 2). As presented and discussed above, the type of instruction, as disclosed by Larsen, differs vastly from one instruction having multiple meanings, as claimed. Thus, Larsen also fails to teach or suggest executing the extended instruction, wherein the portion of the corresponding address determines a meaning for the extended instruction from the possible meanings, as claimed.

Accordingly, Larsen fails to render independent Claim 1, under 35 U.S.C. §103(a). Independent Claims 5 and 10 recite limitations similar to that of

independent Claim 1 and are patentable for similar reasons. Dependent claims are patentable by virtue of their dependency. As such, allowance of Claims 1-3, 5-7, 9-12 and 14 is earnestly solicited.

As per Claims 9 and 14, the rejection asserts that "a compiler by definition translates high-level language into object code prior to the execution of a program. Thus, a compiler generates instructions that are executable on a processor." The rejection further asserts that it is therefore obvious that "this is a semantic rule that the compiler of Larsen must follow and correctly place all type 2 instructions only in memory locations ending with '111' and place all type 1 instructions at other memory locations. Thus, the compiler also stores instructions in memory places." Applicants respectfully disagree with this interpretation in view of the following.

A compiler is a computer program that may translate text written in computer language into another computer language. Text written programs are generally saved in a memory and upon execution of the program, the written program is compiled to generate machine language code. Thus, the mere disclosure of a compiler in Larsen, fails to teach or suggest that generating the instruction and storing the instruction, as claimed, are performed by a compiler.

Claims 4, 8 and 13 are rejected under 35 U.S.C. §103(a) as being allegedly unpatentable over Larsen in view of ("390 Principles of Operation") (hereinafter IBM). Applicants respectfully submit that IBM fails to remedy the failures of Larsen with respect to independent Claims 1, 5 and 10. Claims 4, 8 and 13 depend from independent Claims 1, 5 and 10 respectively. Thus, Larsen alone or in combination with IBM fails to render Claims 4, 8 and 13 obvious. As such, allowance of Claims 4, 8 and 13 is earnestly solicited.

For the above reasons, the Applicants request reconsideration and withdrawal of the rejections of record.

CONCLUSION

In light of the above listed remarks, reconsideration of the rejected Claims 1-14 is requested. Based on the arguments presented above, it is respectfully submitted that Claims 1-14 are in condition for allowance.

Please charge any additional fees or apply any credits to our PTO deposit account number: 50-4160.

Respectfully submitted,
MURABITO, HAO & BARNES LLP

Dated: Dec 17th, 2007

Amir A. Tabarrok
Registration No. 57,137

MURABITO, HAO & BARNES LLP
Two North Market Street
Third Floor
San Jose, California 95113

(408) 938-9060 Voice
(408) 938-9069 Facsimile